

Appl. No. 10/016,933
Replacement Sheet

FIG. 7B

```
* Typically, this class is used to construct parameters for, or interpret
* responses from, methods in the UDDIProxy class.
*
* <p><b>Element description:</b></p>
*
%foreach%annotation% 720
* %annotation%
%end% 722
*
* <p>
*
* @author David Melgar (dmelgar@us.ibm.com)
*/
public class %ElementName% extends UDDIElement {
    public static final String UDDI_TAG = "%elementName%";

    protected Element base = null;

    %ifText%
        String text = null;
    %end%
    %forEach%Attribute%
        String %attribute% = null;
    %end%A
    %forEach%Child%
        %Child% %child% = null;
    %end%
    %forEach%ChildCollection%
        // Vector of %Child% objects
        Vector %child% = new Vector();
    %end%
```

Appl. No. 10/016,933
Replacement Sheet

FIG. 7D

```

/**
 * Construct the object from a DOM tree. Used by
 * UDDIProxy to construct an object from a received UDDI
 * message.
 *
 * @param base Element with name appropriate for this class.
 *
 * @exception UDDIException
 *           Thrown if DOM tree contains a SOAP fault or
 *           disposition report indicating a UDDI error.
 */
public %ElementName%(Element base) throws UDDIException {
    // Check if it is a fault. Throws an exception if it is.
    super(base);
    %ifText%
        text = getText(base); 760
    %end%
    %forEach%Attribute%
        %attribute% = base.getAttribute("%attribute%");
    %end%
    NodeList nl = null;
    %forEach%Child%
        nl = getChildElementsByTagName(base, %Child%.UDDI_TAG);
        if (nl.getLength() > 0) {
            %Child% = new %Child%((Element)nl.item(0));
        }
    %end%
    %forEach%ChildCollection%
        nl = getChildElementsByTagName(base, %Child%.UDDI_TAG);
        for (int i=0; i < nl.getLength(); i++) {
            %child%.addElement(new %Child%((Element)nl.item(i)));
        }
    %end%
}

```